(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
15 March 2001 (15.03.2001)

PCT

(10) International Publication Number
WO 01/18631 A1

(51) International Patent Classification[7]: G06F 1/00

(21) International Application Number: PCT/EP00/08301

(22) International Filing Date: 24 August 2000 (24.08.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
9920644.3 2 September 1999 (02.09.1999) GB

(71) Applicant *(for all designated States except US)*: **MEDICAL DATA SERVICES GMBH** [DE/DE]; An der Alten Ziegelei 20, 48157 Munster (DE).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **ELFERING, Ingo** [DE/US]; SmithKline Beecham Corporation, 709 Swedeland Road, King of Prussia, PA 19406 (US).

(74) Agent: **GIDDINGS, Peter, John**; SmithKline Beecham Corporate Intellectual Property, Two New Horizons Court, Brentford, Middlesex TW8 9EP (GB).

(81) Designated State *(national)*: US.

(84) Designated States *(regional)*: European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published:
— *With international search report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 01/18631 A1

(54) Title: METHOD FOR ANONYMIZING DATA

(57) Abstract: This invention relates to a method for identifying the source or origin of data in databases containing anonymized data. It is a particularly useful means for providing a patient or his or her medical provider with access to patient data over a network where the data is stored in an anonymized data file.

Method for Anonymizing Data

Field of the Invention

This invention relates to a method for identifying the source or origin of data in databases containing anonymized data. It is particularly useful means for providing a patient or his or her medical provider with access to patient data over a network where the data is stored in an anonymized data file.

Background of the Invention

Patient data can normally be stored in a centralised data file such as a central server only if it is adequate secured and anonymized. This normally leads to trusted third party-environments or encryption of the data, or both.

While storing the data in a central archive and protecting it by encryption is one means of insuring patient privacy, encryption of the data is not a viable solution if the data is to be used for any task on the server (e.g. running epidemiological analyses).

Trusted Third Party (TTP) service is a current way for anonymizing patient data. The data is sent to a TTP, which takes the data and replaces all patient identifiers with a new code. The TTP matches codes against the patients – it therefore knows all the codes and patients. This may make it a vulnerable target for hacking. Also the services are complex and therefore expensive. And all data has to be routed through the TTP so it cannot be used or accessed directly.

This invention provides a method for anonymizing patient data without having to use a TTP. In addition, the instant algorithm can be used to anonymize any data source and the data associated with that source in a data base, patient medical data being but one example of the application of this invention.

Summary of the Invention

This invention comprises, in a networked computer system, a method for anonymizing data unique to a data owner in a data file by:

identifying at least one unique alpha-numeric identifier for a data owner having owner-associated data in the data file;

generating a hashcode using a computational device for said unique identifier;

optionally encrypting the hashcode;

linking the hashcode with owner's data;

communicating the code and data to a database; and

providing the data owner or authorized party with computational means for accessing and querying the database over a network; and

on said computational means, a computational method for deciphering the hash-code and associating it with the owner and data transmitted to the computational means from the query made to the database.

Also, this invention represents a secure networked computational system for storing and accessing data by a data owner or an authorized party, the system comprising:

a) a server capable of receiving and storing a hash-code, data associated with that hash-code and linking said hash-code with its associated data;

b) a client networked with or capable of linking up with the server and having on it:

i) a means for generating a hash-code based on a unique identifier associated with the data owner;

ii) a means for communicating said hash-code and associated data to the server;

ii) a means for querying the data on the server for instances of a hash-code and data associated with it and receiving the results of said query; and

iii) a means for decoding the hash-code to re-identify the data owner and associated data.

In yet another aspect, this invention relates to a method for preventing theft and use by a third party of a data owner's data residing on a networked computer-readable medium, which method comprises:

generating a hashcode for an identifier unique to the data owner using a computational device;

optionally encrypting the hashcode;

linking the hashcode with owner's data;

communicating the code and data to a networked database;

providing the data owner or authorized party with computational means for accessing and querying the database over a network; and

on said computational means, a computational method for deciphering the hash-code and associating it with the owner and data transmitted to the computational means from the query made to the database.

Detailed Description of the Invention

In its broadest iteration, this invention provides a means for anonymizing data without the need for a TTP or encryption technologies, particularly in the context of client/server computational system illustrated by the likes of the Internet. The upshot is that one or many parties can generate data for a single data owner and that data owner can search through one or more data files where that data resides and get direct access to his or her data, and be assured of privacy. This allows one to store data for many data owners in or several data bases while permitting secure and private access to the data for each data owner, to the exclusion of the other owners' with data in that data base. Hence the data base can be made available in one or several data bases using a client/server configuration and running over a public network such as the Internet without loss of privacy and the need to

collect the data into one single data base. And this method also provides a means for carrying out analyses of all of the data in the data base without compromising the privacy of any data owner; thus avoiding the problem created if encryption is used as the means for creating a secure link between data owner and data in a networked data file.

For purposes of convenience, this invention is set forth herein after in terms of its application in the context of health care activities and patient privacy. What is set forth herein with regards to the anonymization of patient data applies equally well to any situation where an individual, company, or group owns or has rights to data which it or they wish to hold in anonymity but still have access it over non-restricted computational systems like the Internet.

The initial step, or action is to identify or create a significant data owner identifier like a social security number and generate a hashcode from this data. Hashcodes are mathematical trap-door functions used by digital signatures. The hashcode calculates from the unique identifier a fix length "number". The number is always the same if the same input is given. The security in the hashcodes lies in the fact there is no way to re-identify from the hashcode value the original data.

This approach can therefore be used to replace the patient identifiers in the data with the hashcode. The data can then be used directly. Since every party in the system will always generate the same hashcode for the same patient, data from one patient across several providers still has the same hashcode.

If data in a given database needs be re-identified as to which data belongs to a given patient, then the hashcode is calculated for the patient again and all data bases are searched for that hashcode. Data with it belongs therefore to the patient and can be accessed only by the patient or someone acting by authorisation of the patient. This system can also be used to authorise activities. For example it can be used to permit a patient to generate a refill for a perscription by accessing an electronic form residing in the server's database or linked to it. This form can be authenticated in regards to the medication(s) prescribed by the doctor, which will already be on the database, time to refill, and authentication of the requestor, for example. This is but one additional example of how this invention could be used in the context of providing healthcare over a networked computer system while maintaining patient confidentiality.

Glossary of terms:

The following terms and definitions are used herein after in describing this invention. These definitions are provided for clarity and certainty in defining the invention at the time it was created. While believed to be accurate at the time this invention was made, nomenclature and usage may change with time. These definitions are to be read as representative of their usage in the art at the time the invention was made. Most of these

definitions were obtained from two sources on the world wide web, the PC Webopeadia by internet.com Corp, copyright 1999, which has a URL of http://webopedia.internet.com/ at the time these definitions were obtained and RSA Corporation's FAQ pages at http://www.rsa.com.

HTML: Short for *HyperText Markup Language*, the authoring language used to create documents on the World Wide Web. HTML is similar to SGML (http://webopedia.internet.com/TERM/S/SGML.html) and XML (http://www.oasis-open.org/cover/xmlIntro.html) although it is not a strict subset.

DHTML: Refers to Web content that changes each time it is viewed. For example, the same URL could result in a different page depending on any number of parameters, such as:

1. Geographic location of the reader
2. Time of day
3. Previous pages viewed by the reader
4. Profile of the reader

There are many technologies for producing dynamic HTML, including CGI scripts (http://webopedia.internet.com/TERM/C/CGI.html), Server-Side Includes (SSI), (http://webopedia.internet.com/TERM/S/SSI.html), cookies (http://webopedia.internet.com/TERM/c/cookie.html), Java (http://webopedia.internet.com/TERM/J/Java.html), JavaScript (http://webopedia.internet.com/TERM/J/JavaScript.html), and Active X (http://webopedia.internet.com/TERM/A/ActiveX.html).

When capitalized, *Dynamic HTML* refers to new HTMLextensions that will enable a Web page to react to user input without sending requests to the Web server. Microsoft and Netscape have submitted competing Dynamic HTML proposals to W3C, which is producing the final specification. W3C is short for *World Wide Web Consortium*, an international consortium of companies involved with the Internet and the Web. The W3C was founded in 1994 by Tim Berners-Lee, the original architect of the World Wide Web. The organization's purpose is to develop open standards so that the Web evolves in a single direction rather than being splintered among competing factions. The W3C is the chief standards body for HTTP (hyper text transfer protocol) and HTML.

Smartcard: A small electronic device about the size of a credit card that contains electronic memory, and possibly an embedded integrated circuit. Smart cards containing an IC are sometimes called *Integrated Circuit Cards (ICCs)*.

Smart cards are used for a variety of purposes, including:

1. Storing a patient's medical records
2. Storing digital case

3. Generating network IDs (similar to a token)

COM Object: A model for binary code developed by Microsoft. The Component Object Model (COM) enables programmers to develop objects that can be accessed by any COM-compliant application. An object is, generally, any item that can be individually selected and manipulated. This can include shapes and pictures that appear on a display screen as well as less tangible software entities. In object-oriented programming, for example, an object is a self-contained entity that consists of both data and procedures to manipulate the data. Both OLE (object linking and embedding) and Active X are based on COM. COM specifications can be found at http://www.microsoft.com/com/fnf.asp.

hashing: Producing *hash values* for accessing data or for security. A hash value (or simply *hash*) is a number generated from a string of text. The hash is substantially smaller than the text itself, and is generated by a formula in such a way that it is extremely unlikely that some other text will produce the same hash value. Hashes play a role in security systems where they're used to ensure that transmitted messages have not been tampered with. The sender generates a hash of the message, encrypts it, and sends it with the message itself. The recipient then decrypts both the message and the hash, produces another hash from the received message, and compares the two hashes. If they're the same, there is a very high probability that the message was transmitted intact.

Hashing is also a method of accessing data records. Take for example a list of names:

- John Smith
- Sarah Jones
- Roger Adams

To create an index, called a *hash table,* for these records, you would apply a formula to each name to produce a unique numeric value. So you might get something like:

- 1345873 John smith
- 3097905 Sarah Jones
- 4060964 Roger Adams

Then to search for the record containing *Sarah Jones,* you just need to reapply the formula, which directly yields the index key to the record.

hash function: A *hash function H* is a transformation that takes an input m and returns a fixed-size string, which is called the hash value h (that is, $h = H(m)$). Hash functions with just this property have a variety of general computational uses, but when employed in cryptography, the hash functions are usually chosen to have some additional properties.

The basic requirements for a cryptographic hash function are:
the input can be of any length,

the output has a fixed length,

- $H(x)$ is relatively easy to compute for any given x ,
- $H(x)$ is one-way,
- $H(x)$ is collision-free.

A hash function H is said to be *one-way* if it is hard to invert, where "hard to invert" means that given a hash value h, it is computationally infeasible to find some input x such that $H(x) = h$.

If, given a message x, it is computationally infeasible to find a message y not equal to x such that $H(x) = H(y)$ then H is said to be a *weakly collision-free* hash function.

A *strongly collision-free* hash function H is one for which it is computationally infeasible to find any two messages x and y such that $H(x) = H(y)$.

For more information and a particularly thorough study of hash functions, see B. Preneel. *Analysis and Design of Cryptographic Hash Functions*. Ph.D. Thesis, Katholieke University Leuven, 1993.

The hash value represents concisely the longer message or document from which it was computed; this value is called the message digest. One can think of a message digest as a "digital fingerprint" of the larger document. Examples of well-known hash functions are MD2 and MD5.

Perhaps the main role of a cryptographic hash function is in the provision of message integrity checks and digital signatures. Since hash functions are generally faster than encryption or digital signature algorithms, it is typical to compute the digital signature or integrity check to some document by applying cryptographic processing to the document's hash value, which is small compared to the document itself. Additionally, a digest can be made public without revealing the contents of the document from which it is derived. This is important in digital timestamping where, using hash functions, one can get a document timestamped without revealing its contents to the timestamping service.

Damgård and Merkle greatly influenced cryptographic hash function design by defining a hash function in terms of what is called a compression function. A *compression function* takes a fixed length input and returns a shorter, fixed-length output. Given a compression function, a hash function can be defined by repeated applications of the compression function until the entire message has been processed. In this process, a message of arbitrary length is broken into blocks whose length depends on the compression function, and "padded" (for security reasons) so the size of the message is a multiple of the block size. The blocks are then processed sequentially, taking as input the result of the hash so far and the current message block, with the final output being the hash value for the message.

The best review of hash function techniques is provided by Preneel (B. Preneel. *Analysis and Design of Cryptographic Hash Functions*. Ph.D. Thesis, Katholieke University Leuven, 1993). For a brief overview it will be noted that hash functions are often divided into three classes:

- those built around block ciphers,
- those which use modular arithmetic, and
- those which have what is termed a "dedicated" design.

By building a hash function around a block cipher, a designer aims to leverage the security of a well-trusted block cipher such as DES to obtain a well-trusted hash function. The so-called Davies-Meyer hash function is an example of a hash function built around the use of DES.

The purpose of employing modular arithmetic in the second class of hash functions is to save on implementation costs. A hash function is generally used in conjunction with a digital signature algorithm which itself makes use of modular arithmetic. Unfortunately, the track record of such hash functions is not good from a security perspective and there are no hash functions in this second class that can be recommended for use today.

The hash functions in the third class, with their so-called "dedicated" design, tend to be fast, achieving a considerable advantage over algorithms that are based around the use of a block cipher. MD4 is an early example of a popular hash function with such a design. Although MD4 is no longer considered secure for most cryptographic applications, most new dedicated hash functions make use of the same design principles as MD4 in a strengthened version. Their strength varies depending on the techniques, or combinations of techniques, employed in their design. Dedicated hash functions in current use include MD5 and SHA-1 as well as RIPE-MD (H. Dobbertin, A. Bosselaers, and B. Preneel. RIPEMD-160: A strengthened version of RIPEMD. In *Proceedings of 3rd International Workshop on Fast Software Encryption*, pages 71-82, Springer-Verlag, 1996) and HAVAL (Y. Zheng, J. Pieprzyk and J. Seberry. HAVAL - a one-way hashing algorithm with variable length output. In *Advances in Cryptology Auscrypt '92*, pages 83-104, Springer-Verlag, 1993).

SHA and SHA-1 stand for The Secure Hash Algorithm . It is the algorithm specified in the Secure Hash Standard (SHS, FIPS PUB 180). It was developed by the National Institute of Standards and Technology, a division of the U.S. Department of Commerce. SHA-1 (National Institute of Standards and Technology (NIST). *Announcement of Weakness in the Secure Hash Standard*. May 1994 is a revision to SHA that was published in 1994; the revision corrected an unpublished flaw in SHA. The design of SHA-1 is very similar to the MD4 family of hash functions developed by Rivest. SHA-1 is also described in the ANSI X9.30 (part 2) standard. The algorithm takes a message of less than 2^{64} bits in length and produces a 160-bit message digest. The algorithm is slightly slower than MD5

but the larger message digest makes it more secure against brute-force collision and inversion attacks. SHA is part of the Capstone project.

MD2, MD4 and MD5 are message-digest algorithms developed by Rivest (http://www.rsa.com/rsalabs/faq/html/3-6-6.html). They are meant for digital signature applications where a large message has to be "compressed" in a secure manner before being signed with the private key. All three algorithms take a message of arbitrary length and produce a 128-bit message digest. While the structures of these algorithms are somewhat similar, the design of MD2 is quite different from that of MD4 and MD5. MD2 was optimized for 8-bit machines, whereas MD4 and MD5 were aimed at 32-bit machines. Description and source code for the three algorithms can be found as Internet RFCs 1319 - 1321 (http://www.rsa.com/rsalabs/faq/html/references.html - Kal92), (http://www.rsa.com/rsalabs/faq/html/references.html - Riv92b), and (http://www.rsa.com/rsalabs/faq/html/references.html - Riv92c).

MD2 was developed by Rivest in 1989. The message is first padded so its length in bytes is divisible by 16. A 16-byte checksum is then appended to the message, and the hash value is computed on the resulting message. Rogier and Chauvaud have found that collisions for MD2 can be constructed if the calculation of the checksum is omitted (http://www.rsa.com/rsalabs/faq/html/references.html - RC95). This is the only cryptanalytic result known for MD2.

MD4 was developed by Rivest in 1990. The message is padded to ensure that its length in bits plus 448 is divisible by 512. A 64-bit binary representation of the original length of the message is then concatenated to the message. The message is processed in 512-bit blocks in the Damgård/Merkle iterative structure, and each block is processed in three distinct rounds. Attacks on versions of MD4 with either the first or the last rounds missing were developed very quickly by Den Boer, Bosselaers (B. den Boer and A. Bosselaers. An attack on the last two rounds of MD4. In *Advances in Cryptology Crypto '91*, pages 194-203, Springer-Verlag, 1992) and others. Dobbertin (H. Dobbertin. Alf Swindles Ann. *CryptoBytes*, 1(3): 5, 1995) has shown how collisions for the full version of MD4 can be found in under a minute on a typical PC. In recent work, Dobbertin (Fast Software Encryption, 1998) has shown that a reduced version of MD4 in which the third round of the compression function is not executed but everything else remains the same, is not one-way. Clearly, MD4 should now be considered broken.

MD5 was developed by Rivest in 1991. It is basically MD4 with "safety-belts" and while it is slightly slower than MD4, it is more secure. The algorithm consists of four distinct rounds, which has a slightly different design from that of MD4. Message-digest size, as well as padding requirements, remain the same. Den Boer and Bosselaers have found pseudo-collisions for MD5. More recent work by Dobbertin has extended the techniques

used so effectively in the analysis of MD4 to find collisions for the compression function of MD5. While stopping short of providing collisions for the hash function in its entirety this is clearly a significant step. For a comparison of these different techniques and their impact the reader is referred to MJ.B. Robshaw. On Recent Results for MD2, MD4 and MD5. RSA Laboratories Bulletin No. 4. November 12, 1996.

Van Oorschot and Wiener (P. van Oorschot and M. Wiener. Parallel collision search with application to hash functions and discrete logarithms. In *Proceedings of 2nd ACM Conference on Computer and Communication Security*, 1994) have considered a brute-force search for collisions in hash functions, and they estimate a collision search machine designed specifically for MD5 (costing $10 million in 1994) could find a collision for MD5 in 24 days on average. The general techniques can be applied to other hash functions.

More details on MD2, MD4, and MD5 can be found in B. Preneel 1993 paper (http://www.rsa.com/rsalabs/faq/html/references.html - Pre93 and M.J.B. Robshaw's 1995 article referenced at (http://www.rsa.com/rsalabs/faq/html/references.html#Rob95c).

MIME: Short for *Multipurpose Internet Mail Extensions*, a specification for formatting non-ASCII messages so that they can be sent over the Internet. Many e-mail now support MIME, which enables them to send and receive graphics, audio, and video files via the Internet mail system. In addition, MIME supports messages in character sets other than ASCII.

There are many predefined MIME types, such as GIF graphics files and PostScript files. It is also possible to define your own MIME types.

In addition to e-mail applications, Web browsers also support various MIME types. This enables the browser to display or output files that are not in HTML format.
MIME was defined in 1992 by the Internet Engineering Task Force (IETF) http://webopedia.internet.com/TERM/I/IETF.html. A new version, called *S/MIME*, (Secure MIME) supports encrypted messages.

Unique Identity Data for a Data Owner/Patient

The hashcode is generated from significant information. In Germany the Insurance Company Code Number and the patient's member number in that company are take together to building a string which adds both numbers and separates them via a dot-character. This identifier is then unique for any given patient. Both items can be found on the German KVK Smartcard and are therefore always available. Also the patient knows these numbers and can therefore give them to any party.

The value in this approach is this: If a hacker computed all hashes for all possible values (e.g. on all number permutation of the full item string) he could generate the hash, but this would not help him to identify the patient since he still would have to have access to a system that would allow him to resolve the patient's ID from these numbers. Member ID

and Insurance ID are both numbers with up to 9 figures so a hacker has to calculate 18^10 hashcodes to get the figures that have build of the hashcode.

If this possibility is perceived as a potential problem the hashcode itself could be encrypted with, for example, the public key of the server if sent to the server and with the public key of the user if used by a user. For a fuller explanation of encrpytion technology see Bruce Schneier's, "Applied Cryptography" ISBN 0-47-11709-9 Addison Wesley. Now any hacker could not break the encryption and would therefore not even be able to get to the hashcode. But the system would still have the capability to track patients on the server (since still each encrypted hash would be the same binary blob again because everybody would encrypt with the public key of the server) and the server can re-identify a patient for a user (since he is the only one which can decrypt with his private key and then decrypt again for a given user who would then be the only person who can decrypt again).

In other healthcare systems any other such unique information can be used. What information items are used for generating this identifier or how long this information is, is without any problems for the invention. Anything unique can be chosen, e.g. SSN's, Master Patient-Index-ID-Numbers, Names, etc. The information just has to uniquely describe any given patient and each provider (or user of the system) has to be able to have this information when he wants to encode or decode information for a patient.

Hash Encoding the Data

The unique identifier string is now taken and the hashcode is calculated. Any hashcode can be used by to ensure proper functionality and robustness. One example is SH-1. (Or SHA-1?). The hashcode transforms the string, which can be of any length and can contain any character, numbers or other binary values. The transformed value as outputted by the hashcode is always of a fixed lengths.

The hashing algorithm takes the data and always generates a fixed length bit stream uniquely presenting the data. Even a one-bit change in the data will generate a different hash. No two different pieces of data can produce the same hash value. A bad hashing algorithm would be the sum of all numbers of a give number, like 44 = 4+4=8. This is bad because it generates doubles etc. But a good algorithm like SH1 (?) fulfils the needs of uniqueness of the outcome.

This output value of the SH-1 operation is then encoded via the base64 algorithm so that it contains only normal characters and therefore is easier to transport and handle.

Identification Processes

Patient Data can then be stored in a database where this hash-coded value now identifies the patient. All other patient identifier data can be deleted.

Each provider can send anonymous patient data to a central server with this hash-coded value. Therefore each patient item is identifiable but also all the data about one

patient can be merged on the server since each item from each provider still has the same hash-coded item. This is the preferred approach. This technique makes it possible to have patient data residing on more than one server and be accessible to the patient or her authorised agent via a networked client, i.e., a PC.

If anybody has to re-identify data for a given patient he has to a) know what kind of unique identifier string elements have been used, and b) he has to have this information items from the patient. For example in Germany one could, and probably would, use the Insurance and Member Codes and one would either need to have access to the Smartcard or the patient would need to give the medical provider this information. Also this information is available on medical claim forms, prescriptions, Smartcards, etc..

One can now re-calculate the hashcode for these items and search the database(s) for all items equalling this item. One will find information for about that given patient, and only that patient, and will not be able to view or access data of another patient via that hashcode. This search can be carried out using a PC which has software capable of interacting with the data base engine where the data is stored. One example is a client/server configuration where the data base engine and the data resides on the server and the client, for example a PC, has access to the server over a public or limited-access network. For example a server running under UNIX or Microsoft's NT could have on it a data base such as Oracle's data base engine or be configured as a Web site. The server would be accessible by PC or MacIntosh (the client) over a virtual private network or an intranet, extranet or the Internet. The client software could be a proprietary software package or one could use an Internet browser such as Netscape or Microsoft's Internet Explorer if the server was set up as a Web site. This invention could also be used in the context of a main-frame operation where access is through terminal emulation.

This can also be used in a website where the physician views analytics, e.g. risks in a patient population. The database holds anonymous information about all patients. If she selects any patient in the analysis the system queries her local databases and tries to find the hashcode value from that patient in the local database. If found it means that this physician knows the patient and the site can retrieve the patient's name etc. from the local database and display this instead of the hash value. Therefore the physician would see which patients this is and can act. (not clear on this point)

Example:

Patient Name	Klaus Zoffenpoff
Insurance Number	1234567
Member Number	123456789
Resulting Patient Identifier String	1234567.123456789

Hash Value (SH-1)	$4330BAAE53B55EC753625A63F8FDD11242B3E113
Base64 encoded resulting Patient ID	QzC6rlO1XsdTYlpj+P3REkKz4RM=

Related Standards:

Base64 Encoding: RFC 2045: Multipurpose Internet Mail Extensions (MIME), Part One: Format of Internet Message Bodies

SHA-1 Hash Encoding Standard: http://csrc.nist.gov/fips/fip180-1.txt and http://www.rsa.com/rsalabs/faq/.

Example 1

A server is set up which separates patient data and medical patient data but links both data sources on the server through a unique ID generated on the server. Both data bases have different access/user rights and are only accessible through a COM-object layer that controls access to the database.

The patient database contains a unique patient ID that is generated from unique information associated solely with that patient. In Germany the Insurance Company Code Number and the patient's member number in the company are taken together and are encoded into a fixed binary number by applying a hash-code algorithm (SHA-1) to the data. Then there is applied a base 64 algorithm on top of that. If a unique identifier is not readily available one can create one for patients, e.g., a master patient index.

This hash-code algorithm generates a fixed bit number, e.g. a 160 bit number, that identifies uniquely the patient. Each healthcare provider can transmit anonymous patient data to the central server with just this hash-code identifier number. Each provider can calculate this number. All providers will calculate the same number independently without sharing information beforehand. The database on the server can link all data elements across all providers using this number since all providers will generate the same number for a given patient. The database on the server will not have any of the patient data from which was generated the hash-code number. Therefore by one or more providers can send anonymously lab results, prescriptions, treatment protocols, etc, to one or more databases where the data is linked up, in a given database, with the patient and held anonymously.

In addition, analyses can be performed on the patient's data in the database and can be resent to the healthcare provider via a Web page and still remain anonymous. This is accomplished as follows:

On the client this Web page replaces a piece of HTML (a <DIV> section) by DTHML script code during the load process of the Web page with the patient's real name and other identifying information. This can be done since the client's Web page, (browser?)

queries a file on the client PC or MacIntosh (XXX/IDT/UMOD object) within the script, for the ID (hash-cone number) that has also been stored in the local database. If the script finds the ID, it replaces the DIV section with the actual patient name. If not, it leaves a marker stating that the patient can not be re-identified. Therefore the user sees all her patients and their data without any additional steps.

Example 2

In scenarios where one already have some large pools of anonymous data in this format one can ask a patient if he is willing to give his ID-elements and then one can generate the hashed value and can access the data from this patient; e.g. you get all Rx information and offer a patient to access the database over the web to get refills etc.

For example one can use e-forms document to see how such a service is useful for exchanging electronic prescriptions and how to later on offer per-patient service from the back-end data.

What is claimed is:

1. In a networked computer system, a method for anonymizing data unique to a data owner in a data file by:

identifying at least one unique alpha-numeric identifier for a data owner having owner-associated data in the data file;

generating a hashcode using a computational device for said unique identifier;

optionally encrypting the hashcode;

linking the hashcode with owner's data;

communicating the code and data to a database; and

providing the data owner or authorized party with computational means for accessing and querying the database over a network; and

on said computational means, a computational method for deciphering the hash-code and associating it with the owner and data transmitted to the computational means from the query made to the database.

2. A secure networked computational system for storing and accessing data solely by a data owner or an authorized party, the system comprising:

a) a server capable of receiving and storing a hash-code, data associated with that hash-code and linking said hash-code with its associated data;

b) a client networked with or capable of linking up with the server and having on it:

i) a means for generating a hash-code based on a unique identifier associated with the data owner;

ii) a means for communicating said hash-code and associated data to the server;

ii) a means for querying the data on the server for instances of a hash-code and data associated with it and receiving the results of said query; and

iii) a means for decoding the hash-code to re-identify the data owner and associated data.

3. A method for preventing theft and use by a third party of a data owner's data residing on a networked computer-readable medium, which method comprises:

generating a hashcode for an identifier unique to the data owner using a computational device;

optionally encrypting the hashcode;

linking the hashcode with owner's data;

communicating the code and data to a networked database;

providing the data owner or authorized party with computational means for accessing and querying the database over a network; and

on said computational means, a computational method for deciphering the hash-code and associating it with the owner and data transmitted to the computational means from the query made to the database.

INTERNATIONAL SEARCH REPORT

Internatir Application No
PCT/EP 00/08301

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F1/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	EP 0 884 670 A (INT COMPUTERS LTD) 16 December 1998 (1998-12-16) abstract; figure 1 page 2, line 20 - line 29	1-3
Y	US 5 606 610 A (JOHANSSON JAN) 25 February 1997 (1997-02-25) column 1, paragraph 1 column 1, line 40 - line 67 column 6, paragraph 2 column 6, line 48 -column 7, line 3 figures 1,2	1-3

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
4 December 2000	11/12/2000

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Arbutina, L

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

Internati· Application No
PCT/EP 00/08301

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0884670 A	16-12-1998	NONE	
US 5606610 A	25-02-1997	SE 501128 C	21-11-1994
		AU 671049 B	08-08-1996
		AU 8118394 A	19-06-1995
		BR 9406073 A	12-12-1995
		CA 2153497 A	08-06-1995
		EP 0732014 A	18-09-1996
		FI 953564 A	26-07-1995
		JP 9510305 T	14-10-1997
		NO 952546 A	17-07-1995
		SE 9303984 A	21-11-1994
		WO 9515628 A	08-06-1995